Singleton Acyclic Mechanisms and their Applications to Scheduling Problems[⋆]

Janina Brenner and Guido Schäfer

Institute of Mathematics, Technical University Berlin, Germany
{brenner,schaefer}@math.tu-berlin.de

Abstract. Mehta, Roughgarden, and Sundararajan recently introduced a new class of cost sharing mechanisms called *acyclic mechanisms.* These mechanisms achieve a slightly weaker notion of truthfulness than the well-known Moulin mechanisms, but provide additional freedom to improve budget balance and social cost approximation guarantees. In this paper, we investigate the potential of acyclic mechanisms for combinatorial optimization problems. In particular, we study a subclass of acyclic mechanisms which we term *singleton acyclic mechanisms.* We show that every ρ-approximate algorithm whose induced cost function is *partially increasing* can be turned into a singleton acyclic mechanism that is weakly group-strategyproof and ρ-budget balanced. Based on this result, we develop singleton acyclic mechanisms for parallel machine scheduling problems with completion time objectives, which perform extremely well both with respect to budget balance and social cost.

Keywords: cooperative game theory, mechanism design, cost sharing mechanisms, combinatorial optimization, scheduling problems.

1 Introduction

We consider the problem of designing truthful mechanisms for *binary demand cost sharing games.* We are given a universe U of players that are interested in a common service, and a cost function $C : 2^U \to \mathbb{R}^+$ that specifies the cost $C(S)$ to serve player set $S \subseteq U$. We require that the cost function C is *increasing*, i.e., $C(T) \leq C(S)$ for every $T \subseteq S \subseteq U$, and satisfies $C(\emptyset) = 0$. In this paper, we assume that C is given implicitly by the cost of an optimal solution to an underlying combinatorial optimization problem $\mathcal{P}$. Every player $i \in U$ has a private, non-negative *valuation* v_i and a non-negative *bid* b_i for receiving the service.

A *cost sharing mechanism* M takes the bid vector $\boldsymbol{b} := (b_i)_{i \in U}$ as input, and computes a binary allocation vector $\boldsymbol{x} := (x_i)_{i \in U}$ and a payment vector $\boldsymbol{p} := (p_i)_{i \in U}$. Let S^M be the subset of players associated with the allocation

⋆ This work was supported by the DFG Research Center Matheon "Mathematics for key technologies".

vector $\boldsymbol{x}$, i.e., $i \in S^M$ iff $x_i = 1$. We say that S^M is the player set that receives service. We assume that a cost sharing mechanism complies with the following two standard assumptions: $p_i = 0$ if $i \notin S^M$ and $p_i \leq b_i$ if $i \in S^M$ (*individual rationality*) and $p_i \geq 0$ for all $i \in S^M$ (*no positive transfer*). In addition, the mechanism has to compute a (potentially suboptimal) feasible solution to the underlying optimization problem $\mathcal{P}$ on the player set S^M. We denote the cost of the computed solution by $\bar{C}(S^M)$. M is *β-budget balanced* if $\bar{C}(S^M) \leq \sum_{i \in S^M} p_i \leq \beta \cdot C(S^M)$.

We assume that players act strategically and each player's goal is to maximize his own utility. The *utility* u_i of player i is defined as $u_i(\boldsymbol{x}, \boldsymbol{p}) := v_i x_i - p_i$. Since the outcome $(\boldsymbol{x}, \boldsymbol{p})$ computed by the mechanism M solely depends on the bids $\boldsymbol{b}$ of the players, a player may have an incentive to declare a bid b_i that differs from his valuation v_i. We say that M is *strategyproof* if bidding truthfully is a dominant strategy for every player.

In this paper, we consider *cooperative cost sharing games*, i.e., players are enabled to form coalitions in order to coordinate their bids. A mechanism is *group-strategyproof* if no coordinated bidding of a coalition $S \subseteq U$ can ever strictly increase the utility of some player in S without strictly decreasing the utility of another player in S.

In recent years, considerable progress has been made in devising truthful mechanisms for cost sharing games. Most notably, Moulin [13] proposed a general framework for designing so-called *Moulin mechanisms* that are truthful and (approximately) budget balance. The strength of Moulin's framework lies in the fact that the resulting mechanisms achieve one of the strongest notions of truthfulness, i.e., group-strategyproofness. Most of the mechanisms for cooperative cost sharing games that are currently prevailing in literature are Moulin mechanisms (e.g., [1, 2, 4, 8, 11, 15, 16]).

Recent negative results [1, 2, 9, 11, 15] show that for several fundamental cost sharing games, Moulin mechanisms can only achieve a very poor budget balance factor. This effect is further amplified if approximate social cost is desired as additional objective. The *social cost* [15] of a set $S \subseteq U$ is defined as $\Pi(S) := \bar{C}(S) + \sum_{i \notin S} v_i$. A mechanism M is said to be *α-approximate* if the social cost of the served set S^M satisfies $\Pi(S^M) \leq \alpha \cdot \Pi^*$, where $\Pi^* := \min_{S \subseteq U}(C(S) + \sum_{i \notin S} v_i)$ denotes the optimal social cost.

Very recently, Mehta, Roughgarden, and Sundararajan [12] introduced a more general framework for designing truthful cost sharing mechanisms termed *acyclic mechanisms*. Acyclic mechanisms implement a slightly weaker notion of truthfulness, called *weak group-strategyproofness*, but therefore leave more flexibility to improve upon the approximation guarantees with respect to budget balance and social cost. A mechanism is *weakly group-strategyproof* [5, 12] if no coordinated bidding of a coalition $S \subseteq U$ can ever strictly increase the utility of *every* player in S.

Our Results. In this paper, we investigate the potential of acyclic mechanisms for combinatorial optimization problems, with a particular focus on scheduling problems. Our contributions are threefold:

1. Singleton Acyclic Mechanisms. We study a subclass of acyclic mechanisms that we call *singleton acylcic mechanisms*. We show that a ρ-approximation algorithm for the underlying optimization problem $\mathcal{P}$ yields a singleton acyclic mechanism that is ρ-budget balanced and weakly group-strategyproof if the cost function $\bar{C}$ induced by the approximation algorithm is increasing. In fact, a weaker condition suffices, namely that the induced cost function is only *partially increasing* (definition will be given in Section 3). Our proof is constructive, i.e., we provide a framework that enables to turn any such approximation algorithm into a corresponding singleton acyclic mechanism. We also provide a means to prove approximate social cost for singleton mechanisms that fulfill an additional *weak monotonicity* property.

A direct consequence of this result is that several lower bounds on the budget balance factor for Moulin mechanisms can be overcome by acyclic mechanisms. For example, Graham's *largest processing time* algorithm [7] gives rise to a 4/3-budget balanced acyclic mechanism for $P| |C_{\max}$, improving upon the lower bound of essentially 2 for Moulin mechanisms [1]. Moreover, using the *shortest remaining processing time* rule, we obtain a 2-budget balanced acyclic mechanism for $P|r_i, pmtn| \sum C_i$ [6], and a 1-budget balanced acyclic mechanism for $1|r_i, pmtn| \sum F_i$ [17], both outperforming the lower bounds of $\Omega(n)$ for Moulin mechanisms [2].

2. Singleton Acyclic Mechanisms for Completion Time Scheduling. We develop acyclic mechanisms for completion time scheduling, both with and without release dates and preemption. Namely, we achieve 1-budget balance and 2-approximate social cost for $P| | \sum C_i$, 1.21-budget balance and 2.42-approximate social cost for $P| | \sum w_i C_i$, and 1-budget balance and 4-approximate social cost for $1|r_i, pmtn| \sum C_i$. Not only are these the first cost sharing mechanisms to achieve constant *social cost* approximation factors, but we also overcome the strong lower bound of $\Omega(n)$ on the *budget balance* factor of any Moulin mechanism for all completion time related objectives [2].

3. Approximation Algorithms for Scheduling Problems with Rejection. Every acyclic mechanism that approximates social cost defines an approximation algorithm for the price-collecting variant of the underlying optimization problem. As a by-product of the results mentioned above, we therefore obtain constant approximation algorithms for the respective machine scheduling problems *with rejection*; a formal definition will be given in Section 2.

Previous and Related Work. In the theoretical computer science literature, the cost sharing framework of Moulin [13] has been applied to game-theoretic variants of many classical optimization problems. Recently, the performance of

Moulin mechanisms has also been studied with respect to social cost, e.g. for Steiner tree and forest, facility location, single-source rent-or-buy network design and machine scheduling; see [2, 4, 8, 15, 16]. However, there exist strong lower bounds on the budget balance and social cost approximation factors that are achievable by Moulin mechanisms [1, 2, 9, 11]. Consequently, Mehta, Roughgarden, and Sundararajan introduced a new class of cost sharing mechanisms called *acyclic mechanisms* [12], which they studied mainly in relation to primal-dual algorithms.

Most of the standard parallel machine scheduling settings are well understood. Lenstra [3] proves that the minimum weighted completion time problem $P| \,|\sum_i w_i C_i$ is NP-complete. Graham's rule [10] approximates the problem by a factor of $\frac{1}{2}\cdot(1+\sqrt{2}) \approx 1.21$. For unit processing times or equal weights, Graham's rule delivers an optimal solution.

With release dates and preemption, minimizing the sum of completion times $P|r_i, pmtn|\sum_i C_i$ becomes NP-hard [14]. Only the single machine case is solved optimally by the shortest remaining processing time (SRPT) algorithm [17]. SRPT is a 2-approximation algorithm for the parallel machine case [6].

2 Preliminaries

2.1 Acyclic Mechanisms

The general mechanism design setting that we consider in this paper has been described in the introduction. We briefly review the definition of acyclic mechanisms introduced by Mehta, Roughgarden, and Sundararajan in this section; the reader is referred to [12] for a more detailed description.

An *acyclic mechanism* is defined in terms of a cost sharing method ξ and an offer function τ. A *cost sharing method* $\xi : U \times 2^U \to \mathbb{R}^+$ specifies for every subset $S \subseteq U$ and every player $i \in S$ a non-negative *cost share* $\xi_i(S)$; we define $\xi_i(S) := 0$ for all $i \notin S$. ξ is *β-budget balanced* if for every subset $S \subseteq U$ we have $\bar{C}(S) \leq \sum_{i\in S} \xi_i(S) \leq \beta \cdot C(S)$. An *offer function* τ defines for every subset $S \subseteq U$ and every player $i \in S$ a non-negative *time* $\tau(i, S)$.

The acyclic mechanism $M(\xi, \tau)$ induced by ξ and τ receives the bid vector $\boldsymbol{b}$ as input and proceeds as follows:

1. Initialize $S := U$.
2. If $\xi_i(S) \leq b_i$ for every player $i \in S$, then halt. Output the characteristic vector $\boldsymbol{x}$ of S and payments $\boldsymbol{p} := (\xi_i(S))_{i\in U}$.
3. Among all players in S with $\xi_i(S) > b_i$, let i^* be one with minimum $\tau(i, S)$ (breaking ties arbitrarily).
4. Set $S := S \setminus \{i^*\}$ and return to Step 2.

For a given subset $S \subseteq U$ and a player $i \in S$, we partition the player set S into three sets with respect to the offer time of i: let $L(i, S)$, $E(i, S)$ and $G(i, S)$ be the sets of players with offer times $\tau(\cdot, S)$ strictly less than, equal to, or strictly greater than $\tau(i, S)$, respectively. The following definition is crucial to achieve weakly group-strategyproofness.

Definition 1. *Let ξ and τ be a cost sharing method and an offer function on U. The offer function τ is* valid *for ξ if the following two properties hold for every subset $S \subseteq U$ and player $i \in S$:*

(P1) $\xi_i(S \setminus T) = \xi_i(S)$ *for every subset $T \subseteq G(i, S)$;*
(P2) $\xi_i(S \setminus T) \geq \xi_i(S)$ *for every subset $T \subseteq G(i, S) \cup (E(i, S) \setminus \{i\})$.*

We summarize the main result of Mehta, Roughgarden, and Sundararajan [12] in the following theorem:

Theorem 1 ([12]). *Let ξ be a β-budget balanced cost sharing method on U and let τ be an offer function on U that is valid for ξ. Then, the induced acyclic mechanism $M(\xi, \tau)$ is β-budget balanced and weakly group-strategyproof.*

2.2 Parallel Machine Scheduling

In a parallel machine scheduling problem, we are given a set U of n jobs that are to be scheduled on m identical machines. Every job $i \in U$ has a non-negative *release date* r_i and a positive *processing time* p_i. The release date r_i specifies the time when job i becomes available for execution. The processing time describes the time needed to execute i on one of the machines. We may also associate a non-negative weight w_i with every job $i \in U$. Every machine can execute at most one job at a time. In the *preemptive* setting, the execution of a job can be interrupted at any point of time and resumed later; in contrast, in the *non-preemptive* setting, job interruption is not allowed.

Given a scheduling algorithm ALG, we denote by $C_i^{\text{ALG}}(S)$ the completion time of job $i \in S$ in the schedule for the set of jobs $S \subseteq U$ output by ALG. We omit the superscript ALG if it is clear from the context to which schedule we refer. Depending on the underlying application, there are different objectives for machine scheduling problems. Among the most common objectives are the minimization of the total weighted completion time, i.e., $\sum_i w_i C_i$, and the makespan, i.e., $\max_i C_i$, over all feasible schedules.

In our game-theoretic view of machine scheduling problems, each job is identified with a player who wants his job to be processed on one of the m machines.

Machine Scheduling with Rejection. Consider an arbitrary scheduling problem $\mathcal{P}$ with job set U and objective function C. A natural variant of this problem is as follows. Suppose every job $i \in U$ has a non-negative *penalty* z_i. We now have the option to either schedule a job $i \in U$ and pay its respective contribution to the objective function value, or to not schedule job i an pay its penalty z_i. More formally, the goal is to compute a subset $S \subseteq U$ of jobs such that the overall cost $C(S) + \sum_{i \in U \setminus S} z_i$ is minimized. We call the resulting problem *scheduling problem with rejection.* (Similar variants for network design problems are usually called *price-collecting*; in the scheduling context with due dates, these variants are sometimes called *scheduling with penalties.*)

It is easy to verify that every cost sharing mechanism that approximates social cost by a factor of α defines an α-approximate algorithm for the underlying optimization problem with rejections.

3 Singleton Acyclic Mechanisms

When thinking about acyclic mechanisms and their offer functions, we like to think of *clusters*. By a cluster we mean a maximal set of players that have the same offer time with respect to a set S, i.e., two players i, j are in the same cluster iff $\tau(i, S) = \tau(j, S)$. With this view, it becomes clear to which extent acyclic mechanisms generalize Moulin mechanisms: To one end, if there is only one cluster that contains all players, Definition 1 requires cross-monotonicity, leading to Moulin mechanisms. To the other end, if all clusters are singletons, i.e., every player has a unique offer time, then (P2) of Definition 1 reduces to (P1) and once a cost share is announced to a player, it can never be changed again. Between these two extremes, there is a great range of other acyclic mechanisms.

However, in this section, we concentrate on the subclass of acyclic mechanisms that result from *singleton offer functions*, i.e., offer functions that induce singleton clusters. We call these mechanisms *singleton acylcic mechanisms*, or simply *singleton mechanisms*.

Let τ be a singleton offer function. In the following, we assume that the elements of a subset $S \subseteq U$ are ordered according to non-decreasing offer times, i.e., $S =: \{i_1, \ldots, i_q\}$ with $\tau(i_l, S) < \tau(i_k, S)$ for all $1 \leq l < k \leq q$. Moreover, we define $S_k := \{i_1, \ldots, i_k\} \subseteq S$ as the set of the first $1 \leq k \leq q$ elements in S. We slightly abuse notation and let for every $i \in S$, S_i refer to the set S_k with $i_k = i$. We are particularly interested in singleton offer functions that satisfy the following *consistency property*.

Definition 2. *A singleton offer function τ is called* consistent *if for all subsets $P \subseteq S \subseteq U$, ordered as $P =: \{j_1, j_2, \ldots, j_p\}$ and $S =: \{i_1, i_2, \ldots, i_q\}$, the following holds: If k is minimal with $i_k \notin P$, then $i_l = j_l$ for all $l < k$.*

Let ALG be a ρ-approximate algorithm for $\mathcal{P}$ and let $\bar{C}$ denote the cost function induced by ALG, i.e., $\bar{C}(S)$ is the cost of the solution computed by ALG for player set $S \subseteq U$. We say that $\bar{C}$ is *partially increasing with respect to a singleton offer function* τ if for every $S \subseteq U$ and $i \in S$, we have $\bar{C}(S_i) \geq \bar{C}(S_{i-1})$. The main result of this section is the following:

Theorem 2. *Let* ALG *be a ρ-approximate algorithm for an optimization problem $\mathcal{P}$. If there exists a consistent singleton offer function with respect to which the cost function induced by* ALG *is partially increasing, then there is a singleton acyclic mechanism that is weakly group-strategyproof and ρ-budget balanced.*

Truthfulness and Budget Balance. A singleton offer function τ combined with an approximation algorithm ALG whose cost function is partially increasing with respect to τ give rise to the following natural cost sharing method. Note that the partial increase property ensures that the cost shares are non-negative.

Definition 3. *The cost sharing method ξ induced by* ALG *and a singleton offer function τ is defined as $\xi_i(S) := \bar{C}(S_i) - \bar{C}(S_{i-1})$ for every $S \subseteq U$ and $i \in S$.*

Together with Theorem 1, the following lemma proves Theorem 2.

Lemma 1. *Let τ be a singleton offer function and* ALG *be a ρ-approximate algorithm that is partially increasing with respect to τ. Moreover, let ξ be the cost sharing method induced by* ALG *and τ. Then the following holds:*

1. *ξ is ρ-budget balanced.*
2. *If τ is consistent, then τ is valid for ξ.*

Proof. By definition of ξ, we have $\sum_{i \in S} \xi_i(S) = \sum_{i \in S}(\bar{C}(S_i) - \bar{C}(S_{i-1})) = \bar{C}(S) - \bar{C}(\emptyset) = \bar{C}(S)$ for all $S \subseteq U$, proving that ξ is ρ-budget balanced.

We next show that τ is valid for ξ. Fix $S \subseteq U$ and $i \in S$. Since τ is a singleton offer function, $E(i, S) \setminus \{i\} = \emptyset$, and so (P2) of Definition 1 reduces to (P1). To prove (P1), let $P := S \setminus T$ for some subset $T \subseteq G(i, S)$ and consider the ordered sets $S =: \{i_1, i_2, \dots, i_q\}$ and $P =: \{j_1, j_2, \dots, j_p\}$. Let k be minimal with $i_k \notin P$. Then, by Definition 2, for all $l < k$, $i_l = j_l$ and hence $P_l = S_l$. Since $T \subseteq G(i, S)$, we have $\tau(i, S) < \tau(i_k, S)$. We conclude that $\xi_i(P) = \bar{C}(P_i) - \bar{C}(P_{i-1}) = \bar{C}(S_i) - \bar{C}(S_{i-1}) = \xi_i(S)$. □

From now on, for a consistent singleton offer function τ and an approximation algorithm ALG whose cost function is partially increasing with respect to τ, we call the mechanism $M := M(\xi, \tau)$ the *singleton mechanism induced by* ALG *and τ*. Given an approximation algorithm ALG, we remark that the budget balance factor of M is independent of the consistent singleton offer function used. However, the choice of the singleton offer function may very well influence the social cost of the solution output by the mechanism. Hence, if the cost function induced by ALG is increasing, i.e., $C(T) \leq C(S)$ for all $T \subseteq S \subseteq U$, we can choose τ solely to achieve a good social cost approximation factor. If not, the *no positive transfer* property restricts the choice of τ to offer functions with respect to which $\bar{C}$ is partially increasing.

Social Cost. The social cost analysis of singleton mechanisms can be alleviated if the induced cost sharing method has the following property:

Definition 4. *Let ξ be the cost sharing method induced by* ALG *and τ. We call ξ* weakly monotone *if for all subsets $T \subseteq S \subseteq U$, $\sum_{i \in T} \xi_i(S) \geq \bar{C}(T)$.*

Theorem 3. *Let $M = M(\xi, \tau)$ be the singleton mechanism induced by* ALG *and a consistent singleton offer function τ. Suppose that ξ is weakly monotone. Then, M approximates social cost by a factor of α if*

$$\frac{\bar{C}(S^M \cup S^*)}{C(S^*) + C(S^M \setminus S^*)} \leq \alpha.$$

Proof. We need to upper bound the ratio between the social cost of the set S^M chosen by the mechanism and a set $S^* := \arg\min_{S\subseteq U}(C(S)+\sum_{i\notin S} v_i)$. We have

$$\begin{aligned}\frac{\Pi(S^M)}{\Pi^*} &= \frac{\bar{C}(S^M)+\sum_{i\in S^*\setminus S^M} v_i+\sum_{i\notin S^M\cup S^*} v_i}{C(S^*)+\sum_{i\in S^M\setminus S^*} v_i+\sum_{i\notin S^M\cup S^*} v_i} \le \frac{\bar{C}(S^M)+\sum_{i\in S^*\setminus S^M} v_i}{C(S^*)+\sum_{i\in S^M\setminus S^*} v_i} \\ &\le \frac{\bar{C}(S^M)+\sum_{i\in S^*\setminus S^M} v_i}{C(S^*)+\sum_{i\in S^M\setminus S^*} \xi_i(S^M)} \le \frac{\bar{C}(S^M)+\sum_{i\in S^*\setminus S^M} v_i}{C(S^*)+C(S^M\setminus S^*)}.\end{aligned}$$

Here, the first inequality follows from Fact 1 below. The second inequality holds because $v_i \ge \xi_i(S^M)$ for every player $i \in S^M$. The last inequality follows from weak monotonicity of ξ and the fact that $\bar{C}(S) \ge C(S)$ for every set S.

Without loss of generality, number the players in $S^* \setminus S^M$ in the order in which they were rejected in the course of the mechanism M, i.e., $S^* \setminus S^M =: \{1,\dots,\ell\}$. For every $i \in S^* \setminus S^M$, let R^i be the subset of players in $S^* \cup S^M$ that were remaining in the iteration in which i was removed, i.e., $R^i := S^M \cup \{i, i+1, \dots, \ell\}$. Since i rejected, we have $v_i < \xi_i(R^i)$. Moreover, by definition of the sets R^i and weak monotonicity of ξ, we obtain $\bar{C}(R^i) = \sum_{j\in R^i} \xi_j(R^i) = \xi_i(R^i) + \sum_{j\in R^{i+1}} \xi_j(R^i) \ge \xi_i(R^i) + \bar{C}(R^{i+1})$. Summing over all $i \in \{1,\dots,\ell\}$ yields

$$\sum_{i\in S^*\setminus S^M} v_i \le \sum_{i=1}^{\ell} \left(\bar{C}(R^i) - \bar{C}(R^{i+1})\right) = \bar{C}(S^M \cup S^*) - \bar{C}(S^M).$$

□

The proof of Theorem 3 uses the following fact, which we state without proof.

Fact 1 *For arbitrary real numbers $a \ge b > c > 0$, we have $\frac{a}{b} \le \frac{a-c}{b-c}$.*

4 Completion Time Scheduling

In this section, we study the performance of singleton mechanisms for total completion time objectives. We distinguish between the model with weights, in which all jobs arrive at time zero and no preemption is allowed, and the model in which jobs have release dates and may be preempted.

4.1 Weighted Completion Time

We consider the problem $P| |\sum w_i C_i$ of scheduling a set of jobs $U := [n]$ non-preemptively on m parallel machines such that the total weighted completion time is minimized. Even for the unweighted version, i.e. when $w_i \equiv 1$, no Moulin mechanism can achieve a budget balance factor better than $n/2$ [2]. However, using singleton acyclic mechanisms, we can heavily improve upon this.

Let ρ^{GR} denote the approximation guarantee achieved by Graham's rule, i.e., ordering the jobs by non-increasing weight per processing time w_i/p_i. For

$P|\,|\sum w_iC_i$, the produced schedule is $(1+\sqrt{2})/2 \approx 1.21$-approximate, which is best possible [10]. In the case of one machine, $\rho^{\mathrm{GR}} = 1$. In the unweighted setting, Graham's rule reduces to the *shortest processing time* policy and also delivers an optimal schedule.

Let $M^{wct} := M(\xi, \tau)$ be the singleton mechanism induced by Graham's rule and the offer function τ defined as follows:

Singleton offer function for Graham's rule: Let σ be a non-increasing weight per processing time order on U. If two jobs $i, j \in U$ satisfy $w_i/p_i = w_j/p_j$, we define $\sigma(i) < \sigma(j)$ iff $i < j$. For every subset $S \subseteq U$, let $\tau(\cdot, S)$ be the order on S induced by σ.

One easily verifies that τ is a consistent singleton offer function. We have $\xi_i(S) = \bar{C}(S_i) - \bar{C}(S_{i-1}) = w_iC_i(S)$, where $C_i(S)$ is the completion time of job i in the schedule computed by Graham's rule. Since $w_iC_i(S) \geq 0$, Graham's rule is obviously partially increasing with respect to τ.

Theorem 4. *The singleton mechanism $M^{wct} = M(\xi, \tau)$ induced by Graham's rule and τ is weakly group-strategyproof, ρ^{GR}-budget balanced and approximates social cost by a factor of $2\rho^{\mathrm{GR}}$.*

Proof. It follows from Theorem 2 that M^{wct} is weakly group-strategyproof and ρ^{GR}-budget balanced. It remains to be shown that M^{wct} is $2\rho^{\mathrm{GR}}$-approximate with respect to social cost. To see that the induced cost sharing method ξ is weakly monotone, note that $C_i(S) \geq C_i(T)$ for every $i \in T \subseteq S$. Thus, $\sum_{i \in T} \xi_i(S) \geq \sum_{i \in T} \xi_i(T) = \bar{C}(T)$. The social cost approximation factor now follows from Theorem 3 and Lemma 2 given below. $\square$

Lemma 2. *Let* ALG *be an algorithm for $P|\,|\sum w_iC_i$ with cost function $\bar{C}$. Let A and B be two disjoint sets of jobs. Then, the cost of an optimal schedule for $A \cup B$ can be bounded by $C(A \cup B) \leq 2 \cdot \big(\bar{C}(A) + \bar{C}(B)\big)$.*

Proof. We prove the inequality individually for each machine $\hat{M}$. Consider the jobs $\hat{A} \subseteq A$ and $\hat{B} \subseteq B$ scheduled on $\hat{M}$ in the runs of ALG on A and B, respectively. We denote by c_i the completion time of job i in his respective schedule, i.e. $c_i := \bar{C}_i(A)$ for all $i \in \hat{A}$ and $c_i := \bar{C}_i(B)$ for all $i \in \hat{B}$.

Consider the schedule which processes all jobs in $\hat{A} \cup \hat{B}$ on $\hat{M}$ according to non-decreasing c_i. The completion time of a job $i \in \hat{A}$ in this schedule is $c_i + c_{i^*}$, where i^* denotes the last job in $\hat{B}$ that is processed before i. Since i^* is processed before i, we have $c_i + c_{i^*} \leq 2c_i$. By exchanging the roles of A and B, we can show the same for the completion time of every job $i \in \hat{B}$.

Since the cost of an optimal schedule for $A \cup B$ is at most that of the schedule produced by repeating the above procedure for each machine, we have

$$C(A \cup B) \leq \sum_{i \in A \cup B} w_i \cdot 2c_i = 2 \cdot \Big(\sum_{i \in A} w_ic_i + \sum_{i \in B} w_ic_i\Big) = 2 \cdot \big(\bar{C}(A) + \bar{C}(B)\big).$$

$\square$

The following example shows that our social cost analysis is tight.

Example 1. Consider an instance of $1|p_i = 1|\sum C_i$ on an even number of n jobs with valuations $v_i = i$ for all $i \in [n]$. Assume that M^{wct} orders the jobs according to increasing valuations (we can always enforce this by perturbing the processing times appropriately) and thus accepts all jobs. Then, $\Pi(S^M) = \bar{C}([n]) = n(n+1)/2$.. However, if we exclude the first $n/2$ jobs from the scheduled set, we obtain a social cost of $C([\frac{n}{2}]) + \sum_{i=1}^{n/2} v_i = n(\frac{n}{2}+1)/2 = n(n+2)/4 \geq \Pi^*$, yielding an approximation ratio of essentially 2.

The following theorem comes directly with the proof of Theorem 4 if we identify the players' valuations with the penalties for not scheduling a job.

Theorem 5. *Graham's rule is a* $2\rho^{\text{GR}}$*-approximate algorithm for the weighted completion time scheduling problem with rejection.*

4.2 Completion Time with Release Dates and Preemption

Now, consider the problem $1|r_i, pmtn|\sum C_i$ of scheduling a set of jobs $U := [n]$ on a single machine such that the total completion time is minimized. Each job $i \in U$ has a non-negative release date r_i, and preemption of jobs is allowed. The *shortest remaining processing time* policy (SRPT) delivers an optimal schedule for this problem [17].

We introduce some more notation in order to give a formal definition of SRPT. Let $e_i(t)$ be the amount of time that has been spent on processing job i up to time t. The *remaining processing* time $x_i(t)$ of job i at time t is $x_i(t) := p_i - e_i(t)$. We call a job i *active* at time t if it has been released but not yet completed at this time, i.e., $r_i \leq t < C_i$. Let $A(t)$ be the set of jobs that are active at time t. SRPT works as follows: At any time $t \geq 0$, SRPT schedules an active job $i \in A(t)$ with minimum remaining processing time, i.e. $x_i(t) \leq x_k(t)$ for all $k \in A(t)$. In the following, we assume that SRPT uses a consistent tie breaking rule, e.g., if $x_i(t) = x_k(t)$ for two different jobs i and k, then schedule the one with smaller index. Throughout this section, let $C_i(S) := C_i^{\text{SRPT}}(S)$ for all $S \subseteq U$.

Let $M^{\text{pct}} := M(\xi, \tau)$ be the singleton mechanism induced by SRPT and the following singleton offer function τ:

Singleton offer function for SRPT: For a given subset $S \subseteq U$, let $\tau(\cdot, S)$ be the order induced by increasing completion times of the jobs in S, i.e., $\tau(i, S) < \tau(j, S)$ iff $C_i(S) < C_j(S)$.

The offer function τ is consistent; we defer the proof to the end of this section. Recall that SRPT is an optimal scheduling policy and thus $\bar{C}(S) = C(S)$. We thus have $\xi_i(S) = C(S_i) - C(S_{i-1}) = C_i(S)$, where the latter follows from Lemma 5. Remark that SRPT is partially increasing with respect to τ because $C_i(S) \geq 0$.

Theorem 6. *The singleton mechanism* $M^{\text{pct}} = M(\xi, \tau)$ *induced by* SRPT *and* τ *is weakly group-strategyproof, budget balanced and approximates social cost by a factor of* 4.

Proof. It follows from Theorem 2 that M^{pct} is weakly group-strategyproof and budget balanced. To prove that M^{pct} approximates social cost, we first show that ξ is weakly monotone. Fix some set S and let $T \subseteq S$. Consider the SRPT schedule for S. If we remove from this schedule all jobs in $S \setminus T$, we obtain a feasible schedule for T of cost at most $\sum_{i \in S \setminus T} C_i(S) \geq C(T)$. Since $\xi_i(S) = C_i(S)$, we have weak monotonicity. Now, the bound on the social cost approximation factor follows from Theorem 3, using Lemma 3 given below. $\square$

The following lemma is used to prove the social cost approximation factor.

Lemma 3. *Let* ALG *be an algorithm for* $P|r_i, pmtn|\sum C_i$ *with cost function* $\bar{C}$. *Let* A *and* B *be two disjoint sets of jobs. Then, the cost of an optimal schedule for* $A \cup B$ *can be bounded by* $C(A \cup B) \leq 4 \cdot \big(\bar{C}(A) + \bar{C}(B)\big)$.

Proof. Phillips et al. [14] prove that any preemptive schedule for $P|r_i, pmtn|\sum C_i$ can be turned into a non-preemptive schedule NP with at most twice the cost. With Lemma 2, we obtain $C(A \cup B) \leq 2 \cdot \big(C^{\text{NP}}(A) + C^{\text{NP}}(B)\big) \leq 4 \cdot \big(\bar{C}(A) + \bar{C}(B)\big)$. $\square$

Consistency. In order to prove that τ is consistent, we need some more notation. Consider the SRPT schedule for a set $S \subseteq U$. Let $i, j \in A(t)$ be two jobs that are active at time t. We define $i \prec_t j$ iff either $x_i(t) < x_j(t)$ or $x_i(t) = x_j(t)$ and $i \leq j$. Note that at any point of time t, SRPT schedules the job $i \in A(t)$ with $i \prec_t j$ for all $j \in A(t)$. Thus, if $i \prec_t j$ for some t, then $i \prec_{t'} j$ for all $t' \in [t, C_i)$. We therefore simply write $i \prec j$ iff there exists a time t with $i \prec_t j$. Let $\sigma(t)$ denote the job that is executed at time t in the SRPT schedule for S; we define $\sigma(t) = \emptyset$ if $A(t) = \emptyset$.

Let $j \in S$ be an arbitrary job and consider the time interval $[r_j, C_j)$. We define the set $\mathcal{C}_j$ of jobs that are *competing* with j as

$$\mathcal{C}_j := \{i \in S \setminus \{j\} : [r_i, C_i) \cap [r_j, C_j) \neq \emptyset\}.$$

Note that $j \notin \mathcal{C}_j$. We partition the jobs in $\mathcal{C}_j$ into a set $\mathcal{W}_j$ of *winning jobs* and a set $\mathcal{L}_j$ of *losing jobs* with respect to j: $\mathcal{W}_j := \{i \in \mathcal{C}_j : i \prec j\}$ and $\mathcal{L}_j := \mathcal{C}_j \setminus \mathcal{W}_j$. Intuitively, suppose i and j are both active at some time t. If i is a winning job, then i prevents j from being executed by SRPT. On the other hand, if i is a losing job, then j prevents i from being executed.

We next investigate the effect of removing a job j from S. We use the superscript T if we refer to the SRPT schedule for $T := S \setminus \{j\}$.

Lemma 4. *Consider the two* SRPT *schedules on job sets* S *and* $T := S \setminus \{j\}$. *For every job* $i \in \mathcal{C}_j$ *that is active at time* $t \in [r_j, C_j)$,

$$x_i^T(t) = x_i(t) \text{ if } i \in \mathcal{W}_j \quad \text{and} \quad x_i^T(t) \geq x_j(t) \text{ if } i \in \mathcal{L}_j.$$

Proof. We partition the time interval $[r_j, C_j)$ into a sequence of maximal subintervals $I_1, I_2, \ldots, I_f$ such that the set of active jobs remains constant within every subinterval $I_\ell := [s_\ell, e_\ell)$. We prove by induction over ℓ that the claim holds for every $t \in [r_j, e_\ell)$.

Note that both schedules are identical up to time $r_j = s_1$. If $\sigma(s_1) \neq j$, then both schedules process the same job during I_1 and the claim follows. Suppose $\sigma(s_1) = j$. This implies that $A(s_1) \cap \mathcal{W}_j = \emptyset$ and thus all jobs in $A(s_1) \setminus \{j\} = A^T(s_1)$ are losing jobs. If $A^T(s_1) = \emptyset$, the claim follows. Otherwise, let $k := \sigma^T(s_1)$ be the job that is processed in the schedule for T. Since k is a losing job, we have $x_k^T(s_1) = x_k(s_1) \geq x_j(s_1)$. Since k and j receive the same processing time during I_1 in their respective schedules, the claim holds for all $t \in [r_j, e_1)$.

Now, assume that the claim is true for every $t \in [r_j, e_{\ell-1})$ for some $\ell > 1$. We show that it remains true during the time interval I_ℓ. By the induction hypothesis, $x_i^T(t) = x_i(t)$ for every job $i \in \mathcal{W}_j$ that is active at time $t \in [r_j, e_{\ell-1})$. This implies that a job $j \in \mathcal{W}_i$ is executed at time $t \in [r_j, e_{\ell-1})$ in the schedule for S iff it is executed at time t in the schedule for T. We thus have $A^T(s_\ell) \cap \mathcal{W}_j = A(s_\ell) \cap \mathcal{W}_j$. Moreover, $x_i^T(t) \geq x_j(t)$ for every job $i \in \mathcal{L}_j$ that is active at time $t \in [r_j, e_{\ell-1})$. Since $x_j(t) > 0$ for every $t \in [r_j, C_j)$, every job $i \in \mathcal{L}_j$ that is active at time $t \in [r_j, e_{\ell-1})$ in the schedule for S must also be active at time t in the schedule for T. Thus, $A^T(s_\ell) \cap \mathcal{L}_j = A(s_\ell) \cap \mathcal{L}_j$. We now distinguish two cases:

(i) First, assume $\sigma(s_\ell) =: k \in \mathcal{W}_j$. Job k then has smallest remaining processing time, i.e., $x_k(s_\ell) \leq x_i(s_\ell)$ for all $i \in A(s_\ell)$. We conclude that

$$x_k^T(s_\ell) = x_k(s_\ell) \leq x_i(s_\ell) = x_i^T(s_\ell) \quad \forall i \in A(s_\ell) \cap \mathcal{W}_j = A^T(s_\ell) \cap \mathcal{W}_j$$
$$x_k^T(s_\ell) = x_k(s_\ell) \leq x_j(s_\ell) \leq x_i^T(s_\ell) \quad \forall i \in A(s_\ell) \cap \mathcal{L}_j = A^T(s_\ell) \cap \mathcal{L}_j.$$

Since we assume that SRPT uses a consistent tie breaking rule, this implies that $\sigma^T(s_\ell) = k$ and the claim follows.

(ii) Now, suppose $\sigma(s_\ell) = j$. (Note that $\sigma(s_\ell) \in \mathcal{L}_j$ is impossible.) Then $x_j(s_\ell) \leq x_i(s_\ell)$ for every $i \in A(s_\ell)$ and $A(s_\ell) \cap \mathcal{W}_j = \emptyset$. But then we also have $A^T(s_\ell) \cap \mathcal{W}_j = \emptyset$ and thus $A^T(s_\ell) \subseteq \mathcal{L}_j$. If $A^T(s_\ell) = \emptyset$, the claim follows. Otherwise, let $k := \sigma^T(s_\ell) \in \mathcal{L}_j$ be the job that is executed at time s_ℓ in the schedule for T. Since $x_k^T(s_\ell) \geq x_j(s_\ell)$ and the remaining processing times of k and j in their respective schedules reduce by the same amount during I_ℓ, the claim follows. ☐

Lemma 5. *Let $T \subseteq S \subseteq U$ and consider the* SRPT *schedule for $S \setminus T$. We have:*

1. *$C_i(S \setminus T) = C_i(S)$ for every job $i \in S \setminus T$ with $C_i(S) < C_j(S)$ for all $j \in T$.*
2. *$C_\ell(S \setminus T) \geq \min_{j \in T} C_j(S)$ for every job $\ell \in S \setminus T$ with $C_\ell(S) > C_j(S)$ for some $j \in T$.*

Proof. Consider the SRPT schedule for S and fix two jobs $i, \ell \in S \setminus T$ such that i fulfills $C_i(S) < C_j(S)$ for all $j \in T$, and ℓ does not. Let j be the job in T with largest completion time among all jobs in T. Remove j from S and consider the resulting SRPT schedule for $S \setminus \{j\}$.

By Lemma 4, the completion times of i and all jobs in $T \setminus \{j\}$ in this schedule remain the same, since none of these jobs lose against j by choice of j. We can repeat this argument until all jobs in T are removed from S.

Now, consider the completion time of job ℓ during the same procedure. As long as $C_j(S) > C_\ell(S)$ for the job j that is currently removed, the completion time of ℓ remains equal. As soon as a job j with $C_j(S) < C_\ell(S)$ is removed, the completion time of ℓ may change, but, again by Lemma 4, it stays greater or equal to $C_j(S)$. Hence, we have $C_\ell(S \setminus T) \geq \min_{j \in T} C_j(S)$. □

Lemma 6. *The singleton offer function τ is consistent.*

Proof. Consider two sets $P \subseteq S \subseteq U$, ordered by τ as $P =: \{j_1, j_2, \ldots, j_p\}$ and $S =: \{i_1, i_2, \ldots, i_q\}$. Let k be minimal with $i_k \notin P$. Then, for all $l < k$, we have $i_l \in P$ by minimality of k, and $C_{i_l}(S) < C_{i_k}(S)$ by definition of τ. Also by minimality of k, for all other $i \notin P$, we have $C_{i_l}(S) < C_{i_k}(S) < C_i(S)$. Hence, Lemma 5 proves that $C_{i_l}(S) = C_{i_l}(P)$ for all $l < k$.

For all other jobs $j \in P$, we have $C_j(S) > C_k(S)$ and thus by Lemma 5, $C_j(P) \geq C_k(S) > C_{k-1}(S) = C_{k-1}(P)$. Hence, we have $i_l = j_l$ for all $l < k$. □

Treating the players' valuations as penalties for not scheduling a job gives the following theorem.

Theorem 7. SRPT *is a 4-approximate algorithm for the completion time scheduling problem $1|r_i, pmtn|\sum C_i$ with rejection.*

References

1. Y. Bleischwitz and B. Monien. Fair cost-sharing methods for scheduling jobs on parallel machines. In *Proc. of the 6th Int. Conf. on Algorithms and Complexity*, volume 3998 of *Lecture Notes in Comput. Sci.*, pages 175–186. Springer, 2006.
2. J. Brenner and G. Schäfer. Cost sharing methods for makespan and completion time scheduling. In *Proc. of the 24th Int. Sympos. on Theoretical Aspects of Computer Science*, pages 670–681, 2007.
3. P. Brucker. *Scheduling Algorithms.* Springer, New York, USA, 1998.
4. S. Chawla, T. Roughgarden, and M. Sundararajan. Optimal cost-sharing mechanisms for Steiner forest problems. In *Proc. of the 2nd Int. Workshop on Internet and Network Economics*, pages 112–123. Springer, 2006.
5. N. Devanur, M. Mihail, and V. Vazirani. Strategyproof cost-sharing mechanisms for set cover and facility location games. In *Proc. of the ACM Conference on Electronic Commerce.* ACM Press, 2003.
6. J. Du, J. Y. T. Leung, and G. H. Young. Minimizing mean flow time with release time constraint. *Theoretical Computer Science*, 75(3):347–355, 1990.
7. R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17(2):416–429, 1969.
8. A. Gupta, J. Könemann, S. Leonardi, R. Ravi, and G. Schäfer. An efficient cost-sharing mechanism for the prize-collecting Steiner forest problem. In *Proc. of the 18th ACM-SIAM Sympos. on Discrete Algorithms.* ACM Press, 2007.
9. N. Immorlica, M. Mahdian, and V. S. Mirrokni. Limitations of cross-monotonic cost sharing schemes. In *Proc. of the16th ACM-SIAM Sympos. on Discrete Algorithms*, pages 602–611. ACM Press, 2005.
10. T. Kawaguchi and S. Kyan. Worst case bound of an LRF schedule for the mean weighted flow time problem. *SIAM Journal on Computing*, 15(4):1119–1129, 1986.

11. J. Könemann, S. Leonardi, G. Schäfer, and S. van Zwam. From primal-dual to cost shares and back: a stronger LP relaxation for the Steiner forest problem. In *Automata, Languages and Programming*, volume 3580 of *Lecture Notes in Comput. Sci.*, pages 930–942. Springer, 2005.
12. A. Mehta, T. Roughgarden, and M. Sundararajan. Beyond Moulin mechanisms. In *Proc. of the ACM Conference on Electronic Commerce*, 2007.
13. H. Moulin. Incremental cost sharing: Characterization by coalition strategy-proofness. *Social Choice and Welfare*, 16:279–320, 1999.
14. C. Phillips, C. Stein, and J. Wein. Minimizing average completion time in the presence of release dates. *Math. Program.*, 82:199 – 223, 1998.
15. T. Roughgarden and M. Sundararajan. New trade-offs in cost-sharing mechanisms. In *Proc. of the 38th ACM Sympos. on Theory of Computing*, pages 79–88, 2006.
16. T. Roughgarden and M. Sundararajan. Approximately efficient cost-sharing mechanisms. arXiv report, http://www.arxiv.org/pdf/cs.GT/0606127, June 2006.
17. L. Schrage. A proof of the optimality of the shortest remaining processing time discipline. *Operations Research*, 16:687 – 690, 1968.